Assignment 4 RANSAC and Stitching

Daniel van de Pavert, Damiaan Reijnears, Shuai Wang

September 2020

Introduction

In this assignment we look into the selection of keypoints and using those keypoints to calculate the correct translation such that both pictures have the same orientation. Combining the data of the keypoints and the translation two images can be stitched together. To complete this task we look into the SIFT algorithm and RANSAC. SIFT is used for the selection of the keypoints in both images and RANSAC is used to match the found keypoints.

1 Image Alignment

1.1 Keypoint selection

Many of the methods we have expanded upon in the past three lab reports involved some kind of image feature detection. These image features are of tremendous importance in computer vision, as it is the *first step* in actually *understanding* the image – interpreting what is illustrated on the image. It is therefore also of great importance to be able to recognize (or track) these features across different images which incorporate some sort of similarity between each other (for example, either the background or foreground being roughly the same). From now on, by using the term *keypoints*, we will refer to 'traceable features of importance.' As the algorithm is one of the most influential methods in AI, involves many concepts we have seen in previous labs (as is obvious from the previous paragraph), and tackles key challenges in Computer Vision (as explained, identifying features in an image is a first step towards *understanding*); we decided that it is of great importance to understand the whole algorithm and have thus thoroughly studied the original research paper.

SIFT

Keypoint detection and filtering For feature detection, SIFT uses an alternative to Harris Corner detector (which we have studied in lab 3), other edge detectors based on derivatives (such as the Canny Edge Detector; which we touched upon in literature) and Gabor filters (another edge detector, cleverly

using the mathematical properties of sine waves; which we have chosen to thoroughly study in lab 2). SIFT, as opposed to all the foregoing methods, uses an approach introduced by Mikolajczyk (2002), for which the basis is formed by finding local maxima and minima (corresponding to pixel intensity values) in a *scale space* (a very important concept to computer vision, which we already adequately explained in lab 2, in which an image is essentially analyzed on different scales, achieved by consecutively blurring an image which is first divided into octaves, for the purpose of capturing features 'living in different scales[1]'). This scale space is based on the *Laplacian* (or more precisely, a roughly equivalent approximation in the form of *Difference of Gaussians*), which does function as an *edge detector* (we have already thoroughly examined its inner workings and applications in section 4.3 in lab 2). For the purpose of eliminating exactly these edge features (which are of less interest compared to other types of image features, as later explained by us in lab 3), SIFT does, however, make use of the *Hessian*. The Hessian is a second-order derivative matrix (a 'gradient of the gradient') for a point in the image, and is slightly related (in the sense of a first-order *Taylor expansion*) to the concept of a *second moment matrix*, which we have inspected in lab 3.

Apart from discarding any keypoints other than corners (as already implicitely explained, DoGs have high responses on edges—they are in fact used for this exact purpose—and these edges are then filtered for rejection by using the Hessian), Lowe does reject keypoints below a certain contrast threshold as well (which is calculated using a first-order Taylor expansion).

Rotational invariance Although not explicitely mentioned, the previous paragraph solidly leads to the concept of *scale invariance* (see the paragraph with title 'discussion on SIFT' further in this section). However, it is also of great importance to preserve *rotational invariance*, as camera sensors (and objects themselves) can be photographed from different rotational angles. Lowe (2004) does find a way (albeit somewhat cumbersome) to preserve rotational invariance of keypoints across different images. Rotational invariance is met by a sort-of 'categorical voting' approach: the *gradient* (a vector consisting of all partial derivatives of a pixel intensity value, measured by the *degree of change* in all 'pixel directions,' and thus being directed towards the direction of *steepest (positive) change* with a magnitude proportional to the relative amount of change; extensively used in all previous lab reports for different applications, all of which make use of this same mathematical property) is calculated for every pixel in the three dimensional neighborhood (26 pixels in total) of a selected keypoint, and the directions of these gradients have a *weighted vote* (determined

[1]If convolving an image with a Gaussian Kernel, setting different values for the *scale parameter* σ, which controls the size of the kernel with which the image is essentially *averaged* or *blurred*, causes different features in the image to be emphasized independently from each other. This follows simply from the fact that, as the scale increases, more detail gets lost, while larger features start to appear more obviously – imagine a tree's leaves first disappearing in its branches and essentially—the whole tree—ending up in one single blurry 'blob.'

by the magnitude of the gradient – so that gradients conveying higher amounts of change get emphasized) in the *total direction* of the keypoint (which becomes the final orientation). This method is established by creating 36 bins of 10 degrees each, and summing the magnitudes of each gradient for which the direction corresponds to the rotational range of the bin. The bin with the highest value is then chosen to determine the keypoint orientation (actually, multiple keypoints with different orientations may be generated for the same keypoint if the next largest bin is at least 0.8 times as large as the first bin larger than that bin). The key thing to understand here is that these gradients will always occur with (roughly) the same magnitudes at the same places, whatever the rotation of the image (assuming no explicit change in fore- or background for the exact region for the considered keypoint). If the image is rotated, a keypoint will *still* be generated, only with a different associated rotational bin.

Fingerprinting All keypoints should be captured in a *discriptor*, which should be generated equally for each (same) keypoint under different scales and rotations. (Note that the orientations of the gradients computed in the previous paragraph, should thus have no effect on the generated descriptor.) accomplishes this, again on a somewhat peculiar way, by collecting 8 gradients directions (in 4x4 subregions around the chosen keypoint) in a vector, for which its magnitude becomes the 'DNA' of the keypoint (as a sidenote: that is 4x4x8=128 gradient calculations per keypoint!).

Discussion on SIFT The way Lowe preserves scale invariance is solid: when accentuating an object on consecutively enlarging scales, in the real world, an object's details continuously *dissolve* into a 'larger whole.' Intuitively, you can think of looking at a person's eyebrows, compared to that person's 'whole eye' (or mouth or nose – elements which probably live in the same scale as the eye), then the person's face, then the whole person itself, and then the whole group of people in which the person is standing. The Gaussian blur does *exactly this*, and, as Lowe also applies **interpolation within scales** (essentially performing interpolation across three dimensions), this real-life concept is flawlessly adopted computationally. However, the way in which rotational invariance is accomplished seems (at least for us) a bit 'unnatural:' the bins are discrete and arbitrary and thus (in theory) not completely foolproof, and seem to lack a direction relationship between the implemented computation and the 'real workings of biology.' Some remarks have also been made regarding to robustness of SIFT's rotational invariance[2]. However, a comparitative of SIFT and its variants show that it is at least 'best of its kind' (Wu et al., 2013). A remark can also be made on the way fingerprints are generated – in theory, these fingerprints might not be unique, which can cause problems in SIFT's applications. In this Assignments,taking a random subset (with set size set to 10) of all matching points between is shown as Figure 1.

[2]Quora, *Computer Vision: Is SIFT really rotational invariant?*, `https://www.quora.com/Computer-Vision-Is-SIFT-really-rotational-invariant`, accessed on October 5th, 2020

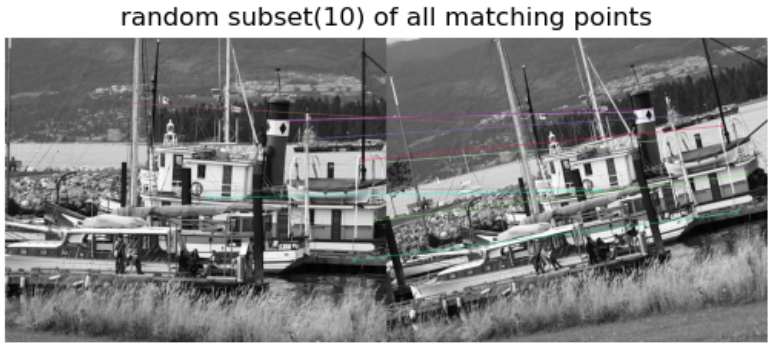

Figure 1: 10 random key points matching between 2 images

1.2 Image transformation

RANSAC

When working with data (or more concretely, when *fitting* data), we often have to deal with *noise*. Sometimes, we do not encounter noise in its exact definition, but we encounter less commonly occurring data points which deviate relatively further from the median – these are called *outliers*. In many applications, it would be ideal to be able to disregard these outlier points (as they are not *describing* the data in its essence). RANSAC is a way to approach this problem: the algorithm (on a bit vulgar way) randomly selects a number of *inliers* and fits a model, then evaluates how well the model does for all other points, and iterates again for a different set of chosen inliers. When *matching* keypoints (obtained by SIFT) across different images, it is often desired to generate a *homography* from one picture to another: an *affine* transformation which essentially relates two pictures in terms of *projection* (intuitively, one could see an image and (a region in) another image as two different geometrical spaces with different bases). The RANSAC algorithm can then be applied on a creative way: by randomly selecting pairs of keypoints (in both pictures), and see how well all other keypoints correspond to that *map* (essentially checking whether other keypoints map to

expected areas), the best pairs (assuming a certain degree of variance influenced by the set number of iterations of RANSAC) of keypoints are selected and then used for the affine transformation between these two pictures. This application of SIFT (and RANSAC) was first proposed by himself (Brown and Lowe, 2003).

This paragraph contains the answers for question 2 of the lab assignment
An affine transformation is a map between two images in terms of stretching/sheering, scaling, translation and rotation (or a combination of all these mentioned transformations) and preservers parallels within the image. As these transformations (depending on its application) might be needed to be expressed in terms of matrix transformations (most often the case in computer graphics, because of the computational advantages, especially related to today's graphical processor units), the concept of *homogeneous coordinates* is introduced. This is an approach in which the dimensionality of an image or space is temporarily expanded with one dimension, where the original dimensions are essentially being reduced to a (hyper)plane (or line) in a higher dimension, where the translation can take place before again reducing the space by one dimension. As already implicated in the previous paragraph, a transformation can be reverse engineered by specifying *pairs of coordinates* in two *existing images* (as opposed to having one image, defining a transformation and then artificially generating a second image). Generally (following the definition of affine transformations stated at the start of this paragraph) an affine transformation needs **three pairs of coordinates**. However, as we are only dealing with scale, translation and orientation when combining two images taken by a camera for the purpose of generating a panorama or mosaic, we need only two pairs of keypoints. Since it also computationally more efficient to initially choose two (instead of three) keypoint pairs (since there is a higher chance of selecting two correct points, compared to selecting three correct points): the number of iterations needed for a certain quality requirement to be met rises exponentially with the number of parameters. Since we apply the *Moore–Penrose* (pseudo-)inverse, we can solve for this *underdetermined* system (normally, an affine transformation defines *six* degrees of freedom, requiring three equations; however, now we will have an underdetermined system of only two equations, but it works as our application only involves mapping *four* degrees of freedom. Images 2-6 shows the result of boat images transform from show the transformations from image1 to image2 and from images2 to image1.

Figure 2: Initial horizontal boat

Figure 3: Transformed slant boat

Figure 4: Initial slant boat

Figure 5: Transformed horizontal boat

One should take note that RANSAC is based on *random sampling*. For every 'full run' of the algorithm, the results will thus be different.

2 Image stitching

2.1 Stitching

Using the keypoints found in both pictures and the optimal transformation matrix the images can be stitched together into one image. This is done by transforming the second image to match the orientation of the base image and finally overlaying the second image on the base image using the keypoints. Following the procedure briefly explained in section 1.2 of this paper, it is worth to note that for every image involved in the stitching process, the image orientation for

the image 'best matching' with all other images is *leading*. Thus, all other images are correspondingly rotated, in order to create the panorama/mosaic. This process leads to a new combined image without the loss of any information. For this part, we combine two images of a train in the street. The applications of this range from the creation of panorama shots, the stabilization of video to medical imaging. Another neat application is multiple-image super-resolution imaging which itself us used in the mapping of outer space, radar and sonar.

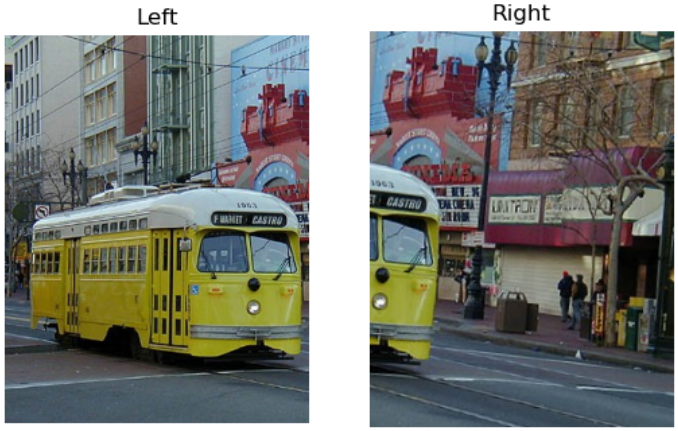

Figure 6: Initial Images

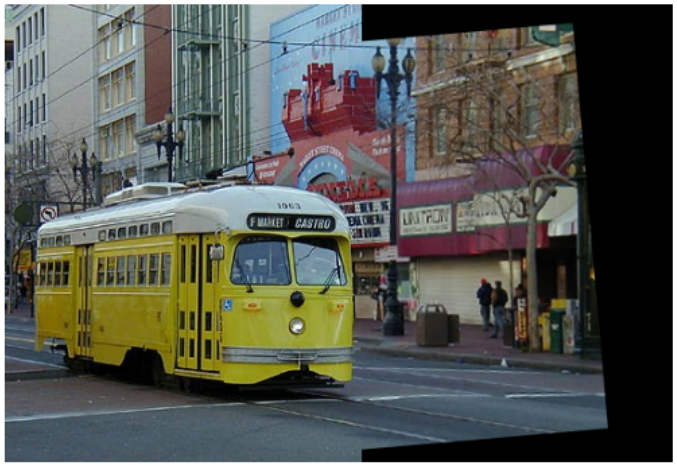

Figure 7: Stitching Result

Conclusion

We utilized the SIFT and RANSAC algorithms to find and connect keypoints in two separate images. The number of keypoints used to get a sufficient match was three. experimented with and we concluded that 3 would lead to a transformation matrix we could use. Using this transformation and the 3 keypoints we could stitch two images together into one frame. Although SIFT does not always correspond to intuition, it produces nice results

References

Brown and Lowe (2003). Recognising panoramas. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1218–1225 vol.2.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110.

Mikolajczyk, K. (2002). *Detection of local features invariant to affines transformations*. PhD thesis.

Wu, J., Cui, Z., Sheng, V. S., Zhao, P., Su, D., and Gong, S. (2013). A comparative study of sift and its variants. *Measurement science review*, 13(3):122–131.